(19) Canadian Intellectual Property Office

An Agency of Industry Canada

Office de la Propri,t, Intellectuelle du Canada

Un organisme d'Industrie Canada

(11) CA 2 351 810 (13) A1

(40) 27.12.2002
(43) 27.12.2002

(12)

(21) 2 351 810

(51) Int. Cl.7: G06F 17/60

(22) 27.06.2001

(71)
MAPFUSION CORP.,
Markham Executive Centre
101 - 200 Town Centre
Boulevard , MARKHAM, O1 (CA).

(72)
UNKNOWN (ZZ).

(74)
GOWLING LAFLEUR HENDERSON LLP

(54) SYSTEME MAPFUSION DE RENSEIGNEMENTS D'AFFAIRES DANS LE DOMAINE SPATIAL
(54) MAPFUSION SPATIAL BUSINESS INTELLIGENCE SYSTEM

Office de la Propriété Intellectuelle du Canada
Un organisme d'Industrie Canada

Canadian Intellectual Property Office
An agency of Industry Canada

CA 2351810 A1 2002/12/27

(21) 2 351 810

(12) DEMANDE DE BREVET CANADIEN
CANADIAN PATENT APPLICATION

(13) A1

(22) Date de dépôt/Filing Date: 2001/06/27

(41) Mise à la disp. pub./Open to Public Insp.: 2002/12/27

(51) Cl.Int.^{7}/Int.Cl.^{7} G06F 17/60

(71) Demandeur/Applicant:
MAPFUSION CORP., CA

(72) Inventeur/Inventor:
UNKNOWN, unknown

(74) Agent: GOWLING LAFLEUR HENDERSON LLP

(54) Titre : SYSTEME MAPFUSION DE RENSEIGNEMENTS D'AFFAIRES DANS LE DOMAINE SPATIAL
(54) Title: MAPFUSION SPATIAL BUSINESS INTELLIGENCE SYSTEM

Introduction

This document provides a detailed description of a unique GIS based business intelligence system delivered via the Internet utilizing thin client technology. The intent of this document is to define the structure, methods, unique user interface items and usage of the system in adequate detail for patent application.

About MapFusion Corp.

MapFusion Corp. is an Application Service Provider (ASP), providing a unique GIS based business solution via the World Wide Web. The ASP service will be provided to clients, who in turn will frame the MapFusion site into their existing Internet or corporate intranet site. As such, MapFusion brings forward an enhancement to clients existing sites, making it appear as if the enhancement is their own.

About the MapFusion System

The MapFusion application is an advanced integration of existing technologies with a unique data model and efficient thin client code. These combined technologies have been engineered to deliver robust GIS business solutions, via the Internet, in a fast, and simple to use format.

The MapFusion system is comprised of four products, which can be combined to provide the level of functionality any client might require. These four products include intelliCarte, intelliCarte Business Tools, RouteLOC and Virtual Business Pages. All of the products operate from the same core components, each product is simply turned on or off to suit the client's needs. The products to be covered under this patent application include intelliCarte, intelliCarte Business Tools, and RouteLOC. The inventors of these three products are (alphabetically) James Bennett, Brian Fleury and Peter Mongrain.

System Overview

The application can be broken into four elemental components:

Data model
Spatial data engine
User interaction engine
Deployment mechanism

The relationship between these four components can be seen in Figure 1.

INSERT FIGURE 1 HERE

While this diagram does not represent a detailed view of the overall system, realizing the elements involved will assist in understanding the relationship of the components which represent the complete system. Each of the individual components are dealt with in greater detail later in this document.

A summary overview of each component is presented here.

Datamodel

The datamodel represents the foundation of the system. It utilizes Oracle (8i) as the database engine. Structrally, it is broken into four major areas:

Supplied Data
Client Data
System Data and Extended Metadata
Code Base

The three data areas house the data utilized by the system, while the code base is a combination of embedded queries and Java code which allow the datamodel to operate as designed.

Spatial Data Engine

The spatial data engine (SDE) resides in the application layer of the system. As with the datamodel it utilizes a third party software, Autodesk's MapGuide as the core engine. The SDE is used to serve the spatial (geographic) information to the end user. Interaction with the SDE is handles by the user interaction engine (UIE).

User Interaction Engine

The user interaction engine (UIE) drives the interface through which the end user accesses the system. All functionality of the UIE is developed in Java, JavaScript, Cold Fusion and HTML. Allaire's Cold Fusion Server pushes servers the user interface to the end user.

Deployment Mechanism

The deployment mechanism is comprised of server and security (i.e. firewall) hardware, software, hosting facilities, bandwidth supply and related support services provided by the hosting datacentre.

Much of the component technology in the deployment mechanism has been developed to interact with, and/or support, and/or deliver, and/or manage the total system. This includes, but is not limited to:

Security modeling
System integration design for overall performance
Scalability and extensibility design
Load management
Communications management

As the overall performance and security of the system represents key components, the technologies integrated into the system become an integral architectural requirement for the successful deployment and use of the system.

Design Intent

This section reviews the development of the concept for the system, and outlines the operational requirements of the system.

History

The concept for the system was derived after research indicated that a truly usable, web enabled and Geographic Information System (GIS) was not available. Further research and discussions indicated that a well designed system could be developed which could be easily deployed to many different marketplaces. The intial research and design commenced in October 1999.

Design Overview

In order to accomplish this a unique datamodel needed to be developed. This datamodel must be adaptable to accomodate the varied data warehousing of unrelated businesses and industries. It must also be able to support extensive metadata and embedded software code, as this will be used to derive the various and unique user interfaces required by each industry and business.

The user interface is programmed in such a way that all buttons, text boxes and other user interaction tools are derived and defined by the datamodel.

As the system will be accessed via both standard "land-line" connections, i.e. telephone, cable modem, DSL, etc., and wireless connections, such as cellular telephone and wireless PDA, close attention is paid to the "footprint" of the application. Most of the operations are performed on the MapFusion servers, with results being served to the end user in a compact manner, thus reducing bandwith requirements. This results in a system which is highly efficient in a wireless connectivity environment.

As the system delivers a highly robust GIS solution to users who may have no formal training in GIS, the user interface and processing systems have been programmed in such a manner as to allow inexperienced users to easily process complex GIS analysis through a series of computer mouse interactions, and by entering simple criteria into text boxes, and/or by selecting available options from list boxes. Users can also interact directly with the spatial infromation contained in the maps contained within the system.

The combination of interactions with both the spatial data and the database data provides the user with a complete analysis result, with a minimum of input on the user side.

Summary Design Specification

The general design specification dictates that a user should be able to get some level of results with a minimum of three "mouse-clicks", and those results should be returned to the user within a maximum of ten seconds, excluding certain multimedia types whose file sizes render a complete return within ten seconds impossible. More refined or detailed results can be achieved with the input of additional criteria, and/or the use of more "mouse-clicks".

These performance specifications are tested using a personal computer, running Microsoft Windows, on an Intel Pentium 120 MHz processor, with 32 megabytes of random access memory (RAM) and a 128 megabyte swap file on the hard drive. Connection to the servers is through a dialup modem, on an analog telephone line, at a connection speed of 28,800 baud.

Ergonomic Design

In order to acheive the goal of simplified user interaction, the system needed to have a front end, or graphical user interface (GUI), designed which allows the user to gain access to the spatial data and the tabular (database) data easily. This front end needed to provide the user with all of the tools required to access and interact with the system, yet be constructed to provide the user with an intuitive, easy to understand and learn, interface.

The system also could not be frustrating to use, in that users, even though accessing the system via the internet, should not need to go through multiple HTML pages of interaction to utilize the system.

The system has been built providing three interfaces: the primary application window, a detailed report window and a detailed data entry window. Each of these windows dynamically adjusts its content based on the rules and metatdata established for each industry, business and client.

Primary Application Window

The primary application window is the only point of interaction with the system, other than the detailed report window. The user will spend most of their time in this window when using the system. This window is designed to be resizable, thus allowing it to scale to full screen size, for standalone operation, or scale to a smaller size if framed into an existing Internet or intranet system.

The primary application window is divided into quadrants, as shown in Figure .

Upper Left Quadrant

The upper left quadrant of the user interface is dedicated to allowing the user to access the spatial data contained in the system. This area is referred to as the Map Window. Through this map window users can easily move and zoom around geographic areas. They can also select items and map features to use for analysis or further query. A customized toolbar is located directly above the map window.

Upper Right Quadrant

The upper right quadrant houses the control tools which allow access to, and interaction with, the database. This area is referred to as the Data Access Window. The very top portion fo this quadrant houses a tabstrip style of toolbar. Selecting an individual tab will present the appropriate user-interaction tools in the remaining area of the quadrant. By default, the user-interaction tools for the first tabstrip are loaded on startup. In some instances, such as when the snapshot report window found in the lower right quadrant is not required, it may stretch down the full height of the screen.

Lower Left Quadrant

The lower left quadrant is used as both a reporting area and as a data entry area. This area is referred to as the Row Report Window. Typically it is the same width as the map window quadrant. In some instances, such as when the snapshot report window found in the lower right quadrant is not required, it may stretch across the full width of the screen. This quadrant provides row/column style reports for the user. Each of the row items has a direct correlation to a positional indicator icon in the map window. For example, houses for sale would be indicated by an icon on the map and will have a corresponding row in the row report. When a row is selected, the corresponding map icon will be highlighted. Likewise, if an icon is selected in the map window, the corresponding row is selected as well. If the snapshot report option is enabled, a snapshot report which corresponds to the selected row will be presented in the lower right quadrant.

The information returned in these row reports are defined initally by the client at time of system setup. Further customization to these reports can be managed directly by the user through the system administration tools. By clicking on the column heading with the mouse, the rows will resort ascending or descending on the values in that column. Each row can have control buttons, selector check boxes and other user interaction tools.

This quadrant can also be used when a number of data entry fields or data interaction functions are required, and would be too crowded in the upper right quandrant, but are not enough to require a seperate window.

Lower Right Quadrant

The lower right quadrant is optional, depending on the requirements of the industry and business. This area is referred to as the Snapshot Report Window. It is used to provide a snapshot style report for an item in the row report. Usually this would contain a picture and some summary information about the row which is currently selected. Some user interaction control functions are also contained in this area, such as the button to launch a detailed report, and a button to zoom in more closely to the selected item on the map.

The layout of the user interface, combined with the fact that the only time a second window is launched is for a detailed report, provides the look and feel of a desktop software application. This look and feel provides end-users, especially those who generally are not comfortable or experienced with the use of Internet applications, with a sense of familiarity, thus providing a degree of reassurance that the application is not "just a web site".

The content of the user interface is defined by the datamodel. Captions for the buttons and icons are defined by rules in the datamodel, and by the contents of the metatdata. The function buttons, icons and links perform when selected by the user are also defined by rules in the datamodel and the contents of the metadata.

Detailed Report Window

The detailed report window is launched from a control button in the row report area or in the snapshot report area. It can also be launched directly from the map window by double-clicking the icon representing the object the user wishes to report on, by selecting the opject in the map window and right-clicking the mouse to bring up the context sensitive menu, or by selecting the object in the map window and clicking the Report icon on the map window toolbar.

The contents, structure and layout of this window is defined by rules and metadata contained in the datamodel. The front end code, comprised of Cold Fusion HTML and JavaScript interacts with code contained in the datamodel to analyze the rules and metadata which defines the report structure for the particular industry, business or client.

Data Entry Window

Akin to the detailed report window is the data entry window. This window is only launched where a significant amount of single record data is required to be entered into the system, which would present the user with a number of fields and options that will not fit into either the upper right or lower left quadrant of the main window.

This data entry window is unique to each industry and client. The code for the window analyses the rules and metatdata contained within the datamodel, then builds the window with the appropriate captions, fields and other data entry tools. The rules, metadatdata and other required information to generate this window during runtime, are defined at time of industry or client setup.

Datamodel Architecture and Operation

The datamodel drives all functionality associated with the system. Built on a third party database application (Oracle 8i), the design incorporates unique table and data structures, code driven rules and metatdata structures, and datawarehousing methodologies.

INSERT DB STRUCT DIAG HERE

As can be seen in the above diagram the datamodel is divided into four major components. Each of these components has been designed for interoperability with the other components of the datamodel, as well as seemless interoperability with the other components which represent the overall system. Many things have been taken into consideration during the design of the datamodel, including, but not limited to, useability, performance, scalability, and security.

All database components are built using object-relational database technologies, instead of standard table-relational technologies. This methodolgy allows the functionality required by a datamodel which needs to support a variety of industries and businesses. Depending on the nature of the data, and it's use within the system, it is stored in a relational object, or in some cases a relational table, and it's structure can be horizontal, similar to a standard database row/column structure, or vertical, in a columnar list type of structure.

All object definitions are managed by the metadata structures within the system.

Supplied Data Area

As the name implies, this area is where the data that MapFusion supplies with the system is housed. This includes spatial and spatial attribute data, demographic data, and business analysis data.

As data is aggregated from a wide variety of suppliers, a standard structure has been defined for each type of data to be stored. Data is then transformed to align to this structure. Aside from developing a consistent structure for ease of integration, the design of this portion of the model has also been optimised for extremely fast access to, and return of, the data. Thus the data structures are combined, related and highly indexed.

Client Data Area

The client data area houses information about the client's business, any users who are permitted to authenticate against the clients profile, and proprietary data entered and managed by the client.

The client's proprietary data can be input directly into the MapFusion datamodel, or can be "connected to" on the client's servers by a variety of mechanisms. Some of these mechanisms include XML and ODBC connectivity tools developed by MapFusion for use with the MapFusion system. In some instances a third party connectivity tool will be used. In each case the client data will be automatically transformed to conform to the requirements of the MapFusion data model.

Metadata Structures

The metadata structures control the data, the object relationships and definitions, as well as many parts of the user interface. Programming controls the metadata, and utilizes the metadata in combination with the information contained in the System Management Data. This programming is described in the section entitled "Code Base".

Some metatdata structures are consistent across all industries and clients, such as system, system data and core element data structure definitions. Others are unique to each industry, and in some cases unique to individual clients. These latter metatdata defintions are contained in both the industry and client specific data tables, and tables which are dedicated to housing industry and client specific metatdata attribute information.

System Management Data

This data area houses the various rules and lookup tables which, when used in conjunction with the metadata structures, define the operation of the system. System management data is comprised of data rules and lookups specific to the operation of the system, regardless of industry or client, as well as those for industry/client specific operation and core supplied data functionality and access.

Code Base

Java code, C++ code, PL/SQL code, and stored procedures are used within the database itself to control data movement in and out of the datamodel. This code is designed specifically to interact with the structure of the datamodel and the metadata contained within the database.

Access to the code base is from the user interface and application layer. All database requests are initiated by the UI and application layer and passed to the code base in the datamodel for processing. Once processing is complete, the results are returned to the calling component, where further post-processing occurs to present the results to the user as appropriate to the initiating request.

Spatial Data Engine Architecture and Operation

The spatial data engine (SDE) utilizes Autodesk's MapGuide® to serve vector-based spatial (mapping) data to the end user.

To provide this data in an efficient and highly usable format, the data is integrated into the system conforming to an exacting engineering standard developed by MapFusion. The vector data format definitions and authoring are managed in this exacting method to insure consistency and performance of data presentation regardless of it's origin.

Authoring the data to achieve this level of consistency and performance involves a combination of scripts to re-engineer the data to a specification developed by MapFusion. Once the scripted process is complete, final presentation of all vector and point data is authored to exacting graphical standards developed by the graphics department of MapFusion.

Spatial served by the system is typically stored in Autodesk SDF format, ESRI SHP format and Oracle 8i Spatial format. By combining these formats the optimal features of each, in terms of data presentation and performance, can be realized.

Request Processing

Requests are posted to the SDE from either, or both, the map window area and/or the data access window.

Requests, or portions of a complete request, orginating from the map window are managed through code, comprised of Java, JavaScript and Cold Fusion. This code, contained in the user interface and application layer, retrieves the necessary spatial information by communicating with the MapGuide client application programming interface (API). It then performs any necessary pre-processing, and passes the request over to the datamodel for processing. If the user has specified criteria, or additional information to be processed through the data access window, this information will be included in the pre-processing stage, and will be handled as is appropriate to the nature of the request. Likewise the request can be generated directly from the data access window, without the user ineracting direcly with the map window.

Once the datamodel has fully processed the request, the results are returned to the user interface and application layer. Any required post-processing of the returned data will be performed at this time. The user interface and application layer code will then post any spatial results to the map window through code which communicates with the MapGuide client API.

Figure ? below demonstrates the flow of information during this process.

INSERT SPATIAL DATA INFORMATION FLOW HERE

User Interaction Engine

The user interaction engine (UIE) is a collection of Java, JavaScript, Cold Fusion and HTML code which is served utilizing Allaire's Cold Fusion Server. This engine is multipurpose in that it not only allows the user to interact with the system, it also is the tool which builds the user interface, based on information and metadata contained within the datamodel.

When a user enters the system, the user interaction engine verifies the user's authentication. Once the user has been identified as a known and registered user ot the system, the UIE reads the datamodel to identify the user's industry and individual business. Once these are known the rules which apply to that user's use of the system are applied to build the appropriate user interface from information contained in data tables, system metadata and extended metadata.

From this point forward, the user can utilize the system as is appropriate for their industry and business, being able to only interact with predefined data sets and system features. This functionality allows users from like industries, but separate businesses, to have access to different data and functionality, in essence a customized system. This allows MapFusion to deploy to many different industries and clients with incurring significant setup costs.

Request Processing

The UIE directly handles all interactions between the user and the system. This includes interaction with the map window, the database and the reporting functions and areas.

Interaction with the map window, as previously defined, is managed with code in the UIE witch communicates directly with the MapGuide API. Through a collection of buttons, text boxes, list boxes and or drop-down list boxes, the user can post queries to the database, and relate those queries directly to the area contained within the map window, if desired.

All other areas of the MapFusion system, which require user interaction, are handled by the UIE. These include the row report, the snapshot report, detailed reports and data entry forms.

Additional User Interface Functions

Different industries can derive different functions from the user interface. These can be automatic functions, such as alerting functions, based on criteria the user establishes, or such things as batch geolocating tools, which can take an existing addressed database the user may have , calculate the latitude and longitude for those addresses and update the database accordingly.

Other tools include, but are not limited to:

Zoom tools – zoom to boundary centre, such as city, address, etc.
Map presentation tools – i.e., change the backdrop data to a different demographic item
Client data management tools – to allow the user to manage their own data
Administrative functions – to allow user's to manage their business and user profiles contained in the system

Operational Description

The system has been designed to provide a consistent "look and feel" regardless of the industry or business using it. The actual user interaction tools and options available to the end-user change based on the users log-in authentication into the system.

User Authentication

When a user logs into the system, the system identifies which functions and features are available to that user. The first level of authentication is handled by the Lightweight Directory Access Protocol (LDAP) server. The LDAP authentication process flags the user's session ID with indicators, which in turn allow the system to identify the level and nature of access the user is entitled to.

Once the user has authenticated through the LDAP server, a combination of code driven by the user interaction engine and code embedded in the database, referenced from this point forward as the "control code", builds the appropriate user interface. The map window is populated with spatial data, covering a geography and scale as defined in the datamodel. The rules in the datamodel also define the layers of spatial data which can be viewed and accessed by the user.

The area of the user interface designed for allowing access to the database is also built by the system at this time. Based on the user's access profile, the control code identifies the tab strip items, control buttons and text/list/dropdown boxes required, and places them in the appropriate areas on the screen.

User Interaction

As the system can be used in different ways by many different industries, it is difficult to establish a standard description of user interaction. However, certain functions and actions remain constant regardless of who is using the system. These functions and actions are described here.

Users will generally define a geographic area they wish to work in, by defining this area in the map window. The system does not require a user to do this, as the spatial, or locational, component of the search can be defined directly in the data access area of the system.

If the user has defined a geographic area using the map window, they can then post queries to the system utilizing the data access area. When posting those queries they can choose to limit the search to the geographic area shown in the map window. The types of queries that can be posted are dependant upon the industry and business profile rules which have been used to build the user interface.

Once the user submits the query(ies) to the system, the map window will present the results graphically, and the row report window and snapshot report window, if enabled, will present summary information to the user. The user can interact with this data either through the map window, the row report window, of the snapshot report window, if enabled. If desired they can also further refine their search, to reduce the number of rows returned. The number of rows of information returned for a given query is defined by the users preferences.

Interaction with the data is provided by several means. If the user passes their mouse cursor over an icon in the map window, the corresponding row in the row report is highlighted. If the snapshot report window is enabled, the information in it will update accordingly. Alternatively, the user can select the desired row from an icon button in the row report, which will highlight the appropriate icon in the map, and update the snapshot report window, if enabled. The user can also zoom in more closely to the item covered in a given row in the row report, by selecting the zoom icon button on the desired row.

More detailed reports on the item in the row report can be launched by clicking on the report icon button in the row report, or on the snapshot report, if enabled.

Certain types of queries generate reports in the row report window, which do not correspond to specific locational icons on the map. These can include driving instructions, demographic reports and other statistical reports, which are suitable for summarization in the row report window. More detailed versions of these reports would be launched in the detailed report window.

Depending on the users login profile, they may have access to additional functions, such as MapFusion's Geolocation tools, additional data sets, map presentation tools, etc.

The user will always have access to the tools necessary to update their personal profile within the system, and, if they have the authority to do so, can edit their businesses profile.

Neural Network Analysis Engine

The Mapfusion system has a neural network analysis engine, which is offered as an add-on component to the MapFusion core product (intelliCarte). This component allows the user to apply advanced network and node tracing to their application.

Applications for this tool include:

MapFusion's RouteLOC vehicle routing and logistics product
tools for telecommunications, allowing users to find the nearest location of a business type (i.e. find the nearest takeout pizza restaurant)
tools for Engineering companies

Each application has a dynamically generated user interface as is appropriate to the use.

The neural network code is written in Java in such a way as to allow process threads to me moved to any CPU in a rack system which has idle time, greatly increasing the overall performance of the application.

One process thread remains running at all times, even when a route or other neural analysis calculation is being processed. This thread is used to test existing routes, based on alerts from the database, against posted changes to the conditions of routes, such as road closure due to accident or weather. If the change condition affects an existing route a series of events takes place. The system recalculates the route around the trouble area. Once the route is recalculated successully, a message is sent to the code base of the MapFusion system. If any monitored vehicles are following the route, and are before the recalculated turn off point, notification procedures, as defined by the end user, are implemented.

Introduction

The MapFusion iButton is a graphical button designed for use in thin client user interfaces. It is unique in both function and size.

The iButton was developed in response to a requirement by MapFusion Corp. for a small footprint, yet highly dynamic and interactive, graphical button for use with the MapFusion Spatial Business Intelligence system.

The inventor of record for the MapFusion iButton is Mark Paterson.

Design Intent

To create a dynamic button without using layers, which can be automatically generated by a database. Thus reducing the overall size of a website or web application dramatically. The filesize (footprint) of the button must be kept to a minimum, to increase efficiencies in bandwidth consumption and connectivity, resulting in performance improvements for thin client applications and Internet HTTP sites which utilize the button.

In addition to being compact, the button must also respond to user interaction. That is, when a user clicks the button, through mouse or keyboard action, it must respond with some level of visually identifiable response, thus creating the illusion that a three dimensional button has been depressed.

iButton Functional Description

Create the illusion that a button is a whole object. Yet the sides of the button do not exist. With not having sides on a button you allow the background colour of the page to show through. This also allows dynamic text of any sort to be placed within the center of the button. This has been accomplished by creating two simple lines with a slight curve on the ends. These are placed on the top and the bottom of the specified dynamic text, which is defined in code or brought in from a database. These two lines are made of opposite colour so that the illusion of light and depth is made.

Design Justification

In order to be truly effective the designed button must meet all of the criteria of the design intent, and must be marketable in demonstrating a true performance and operational benefit when in use.

Standard Button Sizing

An average website or thin client application could contain at least 100 different buttons or more.

These buttons range from an absolute minimum of 255 bytes and up. A separate button is required for each individual state, i.e. over, down, etc.. Ninety percent of all websites have at least 2 states to each button resulting in 200 separate images in total. Many buttons operate in a tristate mode, no-action, mouse over, and mouse down.

Assuming the average site or application has 100 buttons, each of which has two states, the following formula applies:

100 (buttons) x 2 (states) = 200 separate images

Assuming a minimum of 255 bytes per image, the following sizing applies:

200 x 255 = 51,000 (51 kB)

Thus to navigate an average Internet web (http) site, the buttons alone would consume 51 kB of bandwidth. At a normalized dialup network connection speed of 28.8 k baud (28,200 bits per second), this represents (with error checking, etc.) an approximate time consumption of:

([bits per byte] x [total bytes] x [transmission error factor]) / [connection transfer rate]

(8 x 51,000 x 1.1) / 28,800 = 15.6 seconds

Thus it would take approximately 16 seconds to load the buttons found on an average web site.

The above calculations assume a minimum graphical element file size. True averages for images of this type generally are approximately 1,500 bytes (1.5kB). Thus a "real world" example would more likely consume the following, in bandwidth and time:

100 x 2 x 1500 = 300,000 bytes

300,000 x 8 = 2,400,000 bits (2.4 MB)

2,400,000 x 1.1 = 2,640,000

2,640,000 / 28,800 = 91.67

Thus a "real world" load up time would be approximately 92 seconds.

The above sizing calculations are based on the image size of the button only, and do not include any code required to operate the image states or interaction.

iButton Sizing

2 lines per state with 2 states. = 4 lines in total at 255 bytes each.

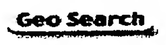
255 x 4 = 1.02 K.

This size will never increase unlike traditional methods, with the amount of buttons used. This is because the same 4 lines are continually being generated from the database. And just different text from the database is applied to the center of each.

This method results in huge file size saving which will increase speeds significantly and save money, by bring the amount of bandwidth needed down tremendously.

Example Button "Up State"

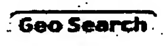
Line 1



Line 2

Example Button "Down State"

Line 3



Line 4

Notice there are no sides to the button but there is the illusion of sides. This is what enables the use of dynamic text in the center with out layering it on top.

Example Operational Code

In order to function properly, the button requires code to populate the caption and to control the animation and appearance. In the example shown below, the caption is updated through the use of hard coded values. In application the caption can be generated from hard coded values, passed to the HTML page as variables, or generated from database queries. This allows one button to be used repeatedly throughout a website, without the need to create multiple graphic images which have captions integrated into them.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<html>
<head>
        <title>Untitled</title>
</head>
<style type="text/css">
<!--
.noboxup {  background-color: #c5d9ff; border: #d6dbe2; font-family: Arial, Helvetica, sans-serif; font-size: 14;
font-weight: bold}
.noboxdn {  background-color: #c5d9ff; border: #d6dbe2; font-family: Arial, Helvetica, sans-serif; font-size: 10;
font-weight: bold}


-->
</style>
<script language="javascript">

top_up = new Image();
top_down = new Image();
bott_up = new Image();
bott_down = new Image();
top_up.src = "topwhite.gif"
top_down.src = "topblackb.gif"
bott_up.src = "btmshade.gif"
bott_down.src = "btmwhiteb.gif"

function dyn1(){
thebutton.value=" THIS IS A"
}
function dyn2(){
thebutton.value=" DYNAMIC"
}
function dyn3(){
thebutton.value=" BUTTON"
}

function butdown(){
bottb.src = bott_down.src;
topw.src = top_down.src;
}

function butoff(){
bottb.src = "btmshade.gif";
topw.src = top_up.src;
}

</script>
<body bgcolor="c5d9ff">
<table width="200" border="0" cellspacing="0" cellpadding="0" align="center">
<tr>
        <td><input type="submit" name="thisis" onClick="dyn1()" value="1">
        <input type="submit" name="thisis" onClick="dyn2()" value="2">
        <input type="submit" name="thisis" onClick="dyn3()" value="3"></td>
</tr>

<tr><td>&nbsp</td></tr>
<tr><td><img src="topwhite.gif" width="82" height="5" border="0" id="topw"></td></tr>
<tr><td align="left" colspan=3><input type="text" name="thebutton" class="noboxup" value="  THIS IS A"
onMouseOver="this.style.cursor='default';butdown();this.style.fontSize = '12'" onMouseOut="this.style.color =
'#000000';this.style.fontSize = '14';butoff()"></td></tr>
<tr><td><img src="btmshade.gif" width="82" height="10" border="0" id="bottb"></td></tr>
<tr>

</tr>
</table>
</body>
</html>
```